ANL-81-39

ANL-81-39

ESTIMATION OF SPARSE JACOBIAN MATRICES AND GRAPH COLORING PROBLEMS

MASTER

by

Thomas F. Coleman and Jorge J. Moré

ARGONNE NATIONAL LABORATORY

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Prepared for the U. S. DEPARTMENT OF ENERGY
under Contract W-31-109-Eng-38

Distribution Category:
Mathematics and Computers
(UC-32)

ANL-81-39

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

ESTIMATION OF SPARSE JACOBIAN MATRICES AND GRAPH COLORING PROBLEMS*

by

Thomas F. Coleman
Jorge J. Moré

Applied Mathematics Division

June 1981

*This work was supported by the Applied Mathematical Sciences Research Program (KC-04-02) of the Office of Energy Research of the U.S. Department of Energy under Contract W-31-109-Eng-38.

TABLE OF CONTENTS

ESTIMATION OF SPARSE JACOBIAN MATRICES AND GRAPH COLORING PROBLEMS

Thomas F. Coleman
Jorge J. Moré

ABSTRACT

Given a mapping with a sparse Jacobian matrix, we investigate the problem of minimizing the number of function evaluations needed to estimate the Jacobian matrix by differences. We show that this problem can be attacked as a graph coloring problem and that this approach leads to very efficient algorithms. The behavior of these algorithms is studied and, in particular, we prove that two of the algorithms are optimal for band graphs. We also present numerical evidence which indicates that these two algorithms are nearly optimal on practical problems.

1. Introduction

Nonlinear problems in optimization and differential equations usually require the estimation of the Jacobian matrix of a mapping $F: \mathbf{R}^n \to \mathbf{R}^m$. For large scale problems, the Jacobian matrix is invariably sparse, and then estimation by differences is attractive because the number of function evaluations needed is often quite small. For example, if the Jacobian matrix is tridiagonal then only three function evaluations are needed. In this paper we consider the problem: Given a mapping $F: \mathbf{R}^n \to \mathbf{R}^m$, what is the number of function evaluations needed to estimate the Jacobian matrix of F?

We assume that it is more efficient to evaluate $F(x)$ than to evaluate separately the components $f_1(x), f_2(x), \ldots, f_m(x)$ of $F(x)$. If this assumption is not satisfied, then it may be desirable to estimate a sparse Jacobian matrix element by element. However, in many applications the components f_i have common sub-expressions, and then it is more efficient to approximate the Jacobian matrix $F'(x)$ by estimating $F'(x)d$ for suitable choices of d. Note that $F'(x)d$ can be estimated, for example, by forward differences,

$$F(x+d) - F(x) = F'(x)d + o(\|d\|) ,$$

or by central differences,

$$\frac{F(x+d)-F(x-d)}{2} = F'(x)d + o(\|d\|^2) .$$

The problem of estimating a Jacobian matrix can thus be formulated in the following terms: For a given m by n matrix A, how many evaluations of Ad are required to uniquely determine all the elements of A?

Curtis, Powell, and Reid [1974] made the crucial observation that if A is sparse then it may be possible to determine A in just a few evaluations of Ad. Moreover, Curtis, Powell, and Reid proposed an algorithm for estimating sparse Jacobian matrices. It turns out that their algorithm -- the CPR algorithm -- can be viewed as a graph coloring algorithm, and that this observation has important consequences. It has led us to investigate the properties of algorithms for estimating sparse Jacobian matrices, and we have shown that it is possible to improve significantly on the CPR algorithm.

Improvements in algorithms for the estimation of sparse Jacobian matrices are important because it is usually necessary to estimate many Jacobian matrices with the same structure. For example, if an improvement allows a Jacobian matrix to be estimated with 3 fewer function evaluations and if the problem (e.g. solution of a system of nonlinear equations or differential equations) requires 100 evaluations of the Jacobian matrix, then an overall improvement of 300 function evaluations is achieved.

In Section 2 we show that it is possible to determine an m by n matrix A by coloring a related graph G(A). We then study some of the special properties of the graph G(A) in Section 3, and in particular, we prove that obtaining an optimal coloring of G(A) is an NP-complete problem. Several graph coloring algorithms are presented in Section 4, and we demonstrate that these algorithms can perform poorly even on relatively simple graphs, and that this behavior is typical of graph coloring algorithms. On the other hand, we prove that the worst possible behavior on graphs of the form G(A) is considerably better than on general graphs. Moreover, in Section 5 we show that two of the algorithms are optimal for band graphs -- a generalization of band matrices. The data structure used to implement the algorithms is described in Section 6, and numerical results for several algorithms are presented in Section 7.

The graph theory necessary to understand this paper is minimal, but for completeness it is presented as needed. We have tried to use the same notation and terminology as Bondy and Murty [1976].

If the Jacobian matrix is symmetric, then the techniques used in this paper can be improved, and we are currently investigating the techniques used by Powell and Toint [1979] for this case.

2. Estimation of Jacobians and Graph Theory

In the introduction we noted that the problem of estimating sparse Jacobian matrices can be formulated as follows: For a given m by n matrix A, obtain vectors $d_1,d_2,\ldots,d_p$ such that $Ad_1,Ad_2,\ldots,Ad_p$ determines A uniquely.

Specifying $Ad_1,Ad_2,\ldots,Ad_p$ gives rise to a system of mp linear equations in the unknowns a_{ij}. If there are τ nonzeroes in A, then we want to obtain the a_{ij}'s by satisfying τ equations. Thus $\tau \leq mp$ and this provides a lower bound on p. A better lower bound can be obtained by noting that each evaluation of Ad only gives rise to one equation in the unknowns in the i^{th} row. If there are ρ_i unknowns in the i^{th} row then we must have $\rho_i \leq p$ and hence,

$$\max\{\rho_i: 1 \leq i \leq m\}$$

is a lower bound on the number of evaluations of Ad needed to determine A.

If the choice of $d_1,d_2,\ldots,d_p$ is such that the system of equations that defines the unknowns can be ordered to diagonal form, then $Ad_1,Ad_2,\ldots,Ad_p$ determines A directly. Equivalently, $Ad_1,Ad_2,\ldots,Ad_p$ determines A directly if for each nonzero a_{ij} there is a d in $\{d_1,d_2,\ldots,d_p\}$ such that

$$a_{ij}\delta_j = (Ad)_i$$

where δ_j is the j^{th} component of d.

It is also possible to determine A indirectly, but this is the subject of future work. In this paper we only consider direct methods for determining A.

Problem. **Obtain vectors $d_1,d_2,\ldots,d_p$ such that $Ad_1,Ad_2,\ldots,Ad_p$ determines A directly with the least value of p.**

To determine an m by n matrix A directly, Curtis, Powell, and Reid [1974] observed that a group of columns can be determined by one evaluation of Ad if no two columns in this group have a nonzero in the same row position. To see this, let $a_1, a_2, \ldots, a_n$ be the columns of A, and let $\{a_j: j \in C\}$ be a group of columns such that no two columns in this group have a nonzero in the same row position. If

$$\delta_j \neq 0 \qquad \text{for} \quad j \in C ,$$

$$\delta_j = 0 \qquad \text{for} \quad j \notin C ,$$

then

$$Ad = \sum_{j \in C} \delta_j a_j ,$$

and it follows that if $a_{ij} \neq 0$ for some $j \in C$ then there is an index i such that

$$(Ad)_i = \delta_j a_{ij} .$$

Thus all the columns a_j for $j \in C$ are determined by Ad.

In this paper we apply the above approach to the problem of estimating sparse Jacobian matrices. This approach can be formalized as follows: A partition of the columns of A is a division of the columns into groups $C_1, C_2, \ldots, C_p$ such that each column belongs to one and only one group. A partition of the columns such that columns in a given group do not have a nonzero in the same row position is consistent with the direct determination of A.

These definitions lead to the main problem of this paper.

Partition Problem. Obtain a consistent partition of the columns of A with the least number of groups.

We are interested in a consistent partition with the least number of groups since each group requires an evaluation of Ad. Stan Eisenstat [1980] has pointed out that in some cases it is possible to determine A directly with less than $\gamma(A)$ vectors where $\gamma(A)$ is the number of groups in an optimal partition. For example, if

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$$

and

$$\chi(A) > \chi(A_1) + \chi(A_2) ,$$

then it is clearly possible to determine A directly with less than $\chi(A)$ vectors. To illustrate this possibility, let

$$A_1 = (D_1 \; D_2)$$

where D_1 and D_2 are nonsingular diagonal matrices of order n, and let

$$A_2 = \begin{pmatrix} z^T & 0 \\ D_3 & B \end{pmatrix}$$

where z is an n-vector with nonzero components, D_3 is a nonsingular diagonal matrix of order n, and B is an n by n matrix with zero diagonal elements and nonzero off-diagonal elements. It is not difficult to show that

$$\chi(A) = 2n , \quad \chi(A_1) = 2 , \quad \chi(A_2) = n ,$$

and thus A can be determined with only n+2 evaluations of Ad.

This example suggests that it may be worthwhile to investigate the general problem of determining A. In this paper we will show, however, that on practical problems the approach based on consistent partitions yields nearly optimal results. This point is discussed further in Section 7.

To attack the problem of minimizing the number of groups in a consistent partition of A, it is advantageous to establish a connection between the partition problem and a graph coloring problem. For this, let us recall some basic graph theory definitions.

A graph G is an ordered pair (V,E) where V is a finite and nonempty set of vertices, and the edges E are unordered pairs of distinct vertices. Thus

$$E \subset \{(u,v):\ u \neq v,\ u,v \in V\} .$$

The vertices u and v are adjacent if (u,v) is an edge with endpoints u and v. The number of edges is denoted by |E|.

A p-coloring of a graph G is a function

$$\phi:\ V \to \{1,2,\ldots,p\}$$

such that $\phi(u) \neq \phi(v)$ if u and v are adjacent. If G has a p-coloring then G is p-colorable and the smallest p for which G is p-colorable is the chromatic number $\chi(G)$ of G. A p-coloring is optimal if $p = \chi(G)$.

A p-coloring ϕ of a graph G = (V,E) induces a partition of the vertices with components $C_1, C_2, \ldots, C_p$ where

$$C_i = \{v \in V:\ \phi(v) = i\} .$$

This suggests that we can associate the partition problem with the coloring of a particular graph. To define this graph, let A be an m by n matrix with columns $a_1, a_2, \ldots, a_n$. The graph G(A) has vertices

$$V = \{a_1, a_2, \ldots, a_n\}$$

and edge (a_i, a_j) if and only if $i \neq j$ and columns a_i and a_j have a nonzero in the same row position. For example, if

$$A = \begin{pmatrix} x & 0 & 0 & 0 & 0 & x & x & x \\ 0 & x & 0 & 0 & x & 0 & x & x \\ 0 & 0 & x & 0 & x & x & 0 & x \\ 0 & 0 & 0 & x & x & x & x & 0 \end{pmatrix}$$

then G(A) is the graph in Figure 2.1.

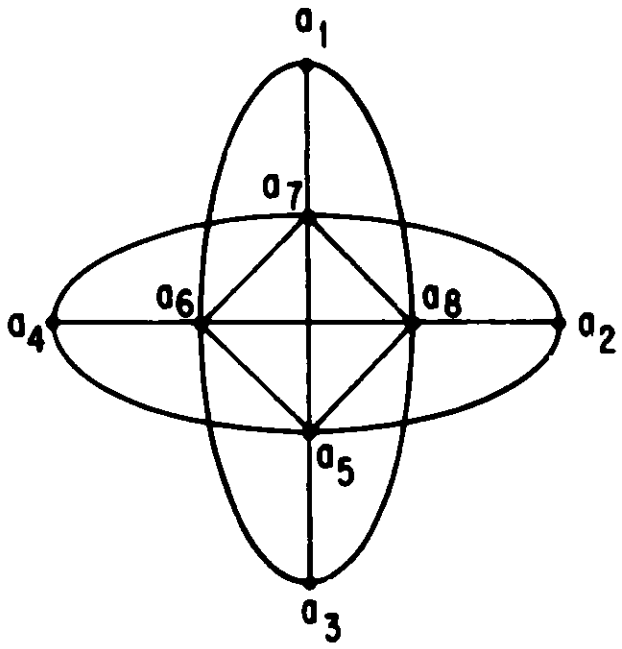

Figure 2.1

The following result is now a direct consequence of the definition of G(A).

Theorem 2.1. ϕ is a coloring of G(A) if and only if ϕ induces a consistent partition of the columns of A.

Once noted, the connection between the partition problem and the graph coloring problem is trivial, and yet it is important because the structure of the graph G(A) is invariant under row and column permutations of A, and thus it is easier to visualize algorithms for the graph coloring problem than for the partition problem.

In view of Theorem 2.1 the partition problem is equivalent to the following problem.

Graph Coloring Problem. Obtain an optimal coloring for G(A).

To illustrate the connection between estimation of Jacobian matrices and graph coloring problem, consider tridiagonal matrices. The graph G(A) for a tridiagonal matrix of order 10 is shown in Figure 2.2, and it is clear that in general the chromatic number of G(A) is 3. It follows that A can be

determined with 3 evaiuations of Ad. In fact, if d only has nonzero components for those vertices (columns) of G(A) with a given color, then all the columns with this color can be determined by evaluating Ad.

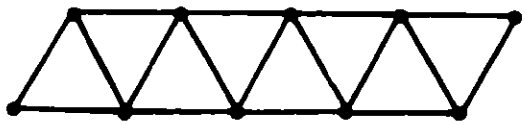

Figure 2.2

We conclude this section by discussing the connection between G(A), adjacency graphs, and intersection graphs. We begin by proving that the adjacency graph of the symmetric matrix A^TA is isomorphic to G(A).

If B is a symmetric matrix then the adjacency graph of B has vertex set

$$\{1,2,\ldots,n\}$$

and edge set

$$\{(i,j): \ i \neq j \ \text{ and } \ b_{ij} \neq 0\} \ .$$

Two graphs $G_1 = (V_1,E_1)$ and $G_2 = (V_2,E_2)$ are isomorphic if there is a bijective function $\psi: V_1 \to V_2$ such that

$$(u,v) \in E_1 \iff (\psi(u),\psi(v)) \in E_2 \ .$$

Theorem 2.2. If A is a nonnegative m by n matrix then G(A) is isomorphic to the adjacency graph of A^TA.

Proof: Let $B = A^TA$. Then

$$b_{ij} = \sum_{k=1}^{m} a_{ki}a_{kj} \ ,$$

and since A is nonnegative, $b_{ij} \neq 0$ if and only if $a_{ki} \neq 0$ and $a_{kj} \neq 0$ for some index k. Thus i and j are adjacent in the adjacency graph of A^TA if and only if a_i and a_j are adjacent in G(A). This establishes our result.

Note that there is no loss of generality in assuming that A is non-negative because only the sparsity pattern of A is relevant to the solution of the partition problem.

It is also worthwhile noting that the construction used to define G(A) is associated with intersection graphs in the graph theory literature (e.g. Harary [1969], page 19). Given a family $F = \{S_1, S_2, \ldots, S_n\}$ of nonempty subsets of

$$S = \bigcup_{i=1}^{n} S_i ,$$

the intersection graph of F has vertex set F and edge set

$$E = \{(S_i, S_j): i \neq j \text{ and } S_i \cap S_j \text{ not empty}\} .$$

It is clear that if

$$S_j = \{i: \ a_{ij} \neq 0\}$$

then G(A) is isomorphic to the intersection graph of this family.

3. Properties of the Graph Coloring Problem

We have related the problem of determining a sparse matrix A to a graph coloring problem. In view of this relationship, it is important to study the graph coloring problem to see if any particular features are present which may facilitate the solution of this problem. In this section we investigate two related questions:

1. Given a graph G with n vertices and a positive integer m, is there an m by n matrix A such that G(A) is isomorphic to G?

Since isomorphic graphs have the same structure, an appropriate answer to this question will allow us to relate properties about general graphs to properties about graphs of the type G(A).

2. Is the problem of determining the chromatic number of G(A) NP-complete?

We refer to Garey and Johnson [1979] for an excellent introduction to the theory of NP-complete problems. The crux of the matter is that NP-complete problems are difficult, and it is believed that NP-complete problems can only be solved by exponential algorithms.

To motivate our answer to the first question, consider the graphs in Figure 3.1.

Figure 3.1

It is not difficult to show that there is an m by 5 matrix A such that G(A) is isomorphic to G_1 if and only if $m \geq 6$. For G_2 it is only necessary to require that $m \geq 3$. The reason is that G_1 can be covered by 6 lines while G_2 can be covered by 3 triangles. The following definitions generalize the notions of lines and triangles.

A graph $G_0 = (V_0,E_0)$ is a subgraph of the graph $G = (V,E)$ if $V_0 \subset V$ and $E_0 \subset E$. Given a nonempty subset V_0 of V, the subgraph $G_0 = (V_0,E_0)$ is induced by V_0 if

$$E_0 = \{(u,v): \ (u,v) \in E, \ u,v \in V_0\} \ .$$

If each pair of distinct vertices in V_0 are adjacent then G_0 is a complete subgraph or clique of G.

Some authors (including Bondy and Murty [1976]) use the term clique to refer to the vertex set of a complete subgraph, but we prefer the above terminology.

The following simple result shows that cliques are important in graph coloring problems.

Lemma 3.1. A lower bound for the chromatic number of a graph G is the size of any clique of G. In particular, if A is an m by n matrix and if ρ_i is the number of nonzeroes in row i then

$$\max\{\rho_i: \ 1 \leq i \leq m\} \leq \chi[G(A)] \ .$$

Proof: If $G_0 = (V_0,E_0)$ is a clique of G then vertices in V_0 are pairwise adjacent, and hence the chromatic number of G is at least the number of vertices in V_0. To complete the proof note that the subset

$$V_i = \{a_j: \ a_{ij} \neq 0\}$$

induces a clique in G(A) with ρ_i vertices.

Definition. A graph G = (V,E) is covered by p cliques $G_1,G_2,\ldots,G_p$ if

$$E = \bigcup_{i=1}^{p} E_i$$

where E_i is the edge set of G_i.

To illustrate this definition consider the graph in Figure 3.2.

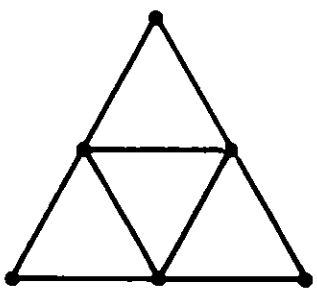

Figure 3.2

This graph has 4 cliques but it can be covered by 3 cliques. In general, a graph may have an exponential number of cliques (Moon and Moser [1965]), but since each edge is a clique, a graph can always be covered by $n(n-1)/2$ cliques.

The next result answers the first question of this section.

Theorem 3.2. Given a graph G with n vertices and a positive integer m, there is an m by n matrix A such that G(A) is isomorphic to G if and only if G is covered by m cliques.

Proof: We first show that for any m by n matrix A, the graph G(A) can be covered by m cliques. For this note that the subset

$$V_i = \{a_j: \ a_{ij} \neq 0\}$$

induces a clique of G(A), provided V_i is not the null set. Moreover, if (a_i,a_j) is an edge of G(A) then columns a_i and a_j have a nonzero in some row k, and thus a_i and a_j are in V_k. It follows that G(A) is covered by the cliques induced by $V_1,V_2,\ldots,V_m$, and thus if G is isomorphic to G(A) then G can also be covered by m cliques.

Conversely, assume that G is a graph with n vertices $v_1,v_2,\ldots,v_n$, and that G can be covered by m cliques $G_1,G_2,\ldots,G_m$. To complete the proof, define the m by n matrix A by setting

$$a_{ij} = 1 \quad \text{if} \quad v_j \in V_i ,$$
$$a_{ij} = 0 \quad \text{if} \quad v_j \notin V_i ,$$

where V_i is the vertex set of G_i and verify that G(A) and G are isomorphic.

We have already noted that the smallest number of cliques which cover a graph G is at most $n(n-1)/2$. This bound is almost optimal. If we consider the graph with vertices $v_1,v_2,\ldots,v_n$ and edge set

$$E = \{(v_i,v_j): \ i \text{ even}, j \text{ odd}\} ,$$

then the smallest number of cliques which cover this graph is $\lfloor n^2/4 \rfloor$. It is not difficult to show (by induction) that a graph with n vertices can always be covered by $\lfloor n^2/4 \rfloor$ cliques and thus this bound is optimal.

This discussion shows that the graphs G(A) where A is an m by n matrix with m less than $\lfloor n^2/4 \rfloor$ are special; in particular graphs G(A) with A a square matrix of order $n > 2$. Thus it may be possible to obtain an algorithm which produces an optimal coloring for these graphs in a polynomial amount of time. The next result shows, however, that this possibility is unlikely, even if we restrict our attention to graphs G(A) where A is a square matrix with order n.

Theorem 3.3. If a graph G with n vertices is covered by m cliques, then there is a square matrix with

$$\text{order } (A) = \max(n, 2m-n)$$

such that G is p-colorable with $p \geq 3$ if and only if G(A) is p-colorable.

Proof: Let $G = (V,E)$ have vertices $v_1, v_2, \ldots, v_n$ and define a graph G_k with vertex set

$$V_k = V \cup \{u_1, u_2, \ldots, u_k\} \cup \{w_1, w_2, \ldots, w_k\}$$

and edge set

$$E_k = E \cup \{(u_i, v_i), (v_i, w_i), (w_i, u_i): \ 1 \leq i \leq k\} .$$

For k = 3, this construction is illustrated in Figure 3.3 for a graph G with 6 vertices which can be covered by 9 cliques.

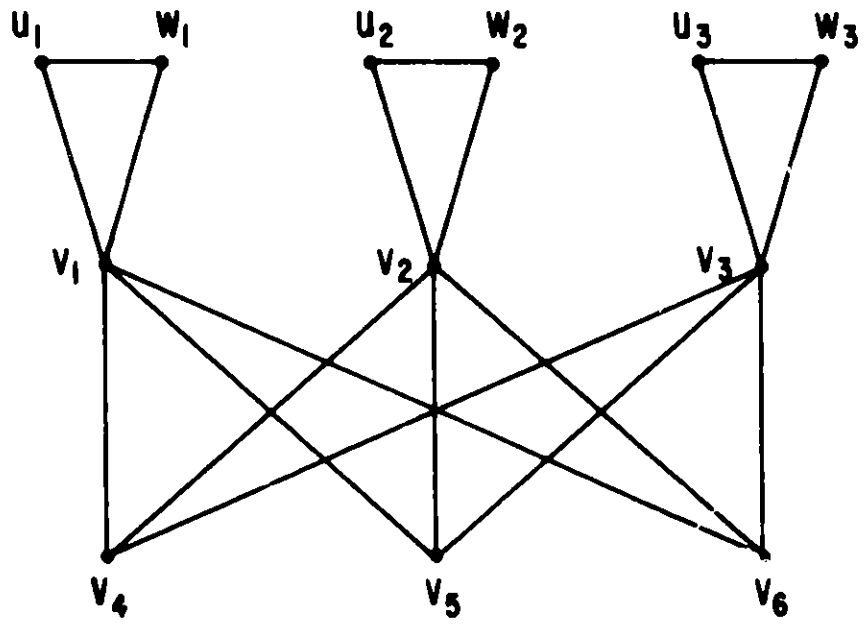

Figure 3.3

Note that in general G_k has $n+2k$ vertices and can be covered by $m+k$ cliques. Moreover, since the additional vertices u_i, w_i only require 2 colors, it is clear that G is p-colorable with $p \geq 3$ if and only if G_k is p-colorable. To complete the proof we show that G_k is isomorphic to an appropriate G(A). For this, note that if

$$k = \max\{0, m-n\}$$

and $\ell = 2k+n$ then G_k has ℓ vertices and can be covered by ℓ cliques. Theorem 3.2 now shows that there is a matrix of order ℓ such that G(A) is isomorphic to G_k, and this completes the proof.

Since a graph G can be covered by $\lfloor n^2/4 \rfloor$ cliques, Theorem 3.3 shows that it is possible to choose a matrix A of order n^2 such that G(A) is isomorphic to G. Thus, if an algorithm requires f(n) operations to produce an optimal coloring of any graph G(A) where A is a square matrix of order n, then this algorithm can produce an optimal coloring for any graph G with n vertices in $f(n^2)$ operations. Since the general graph coloring problem is NP-complete, it follows that the partition problem for square matrices is NP-complete, and thus optimal algorithms for this problem are likely to be exponential. What

is needed is an algorithm which is optimal or nearly optimal in practice; this is the subject of the next section.

It is also worthwhile noting that even restricted graph coloring problems can be NP-complete. For example, the following two problems are known to be NP-complete.

a) Given a graph G, decide if G is 3-colorable.

b) Given a graph G, decide if G is p-colorable with

$$p \leq \alpha_1 \chi(G) + \alpha_2$$

for some constants α_1 and α_2 with $\alpha_1 < 2$.

For references to the literature on NP-complete graph coloring problems, consult Garey and Johnson [1979].

4. Graph Coloring Algorithms

The literature on graph coloring algorithms is extensive; see Brélaz [1979] for some references and recent work. However, some of the literature is not relevant. For example, algorithms for coloring planar graphs are not suitable because if a row of A contains 5 nonzero elements then G(A) has a clique on 5 vertices, and then Kuratowski's theorem (e.g., Bondy and Murty [1976]) implies that G(A) is not planar. The graph coloring algorithms which are most likely to be useful are those designed for general graphs. On the other hand, the computational results available for these algorithms have not been helpful because they are usually restricted to graphs with at most 100 vertices, and in our applications we are interested in much larger graphs.

We are interested in coloring algorithms for G(A) which run in time proportional to

$$\sum_{i=1}^{m} \rho_i^2 ,$$

where ρ_i is the number of nonzero elements in the i^{th} row of A; algorithms for coloring G(A) which run in time proportional to n^2 are not acceptable. This requirement eliminates many coloring algorithms.

We now present a class of graph coloring algorithms and investigate some of the members of this class.

Algorithm. Let G = (V,E) be a graph with the vertices ordered $v_1,v_2,\ldots,v_n$. For k = 1,2,...,n the sequential algorithm assigns v_k to the smallest possible color.

For graphs of the form G(A) it is natural to consider the vertices in the order $a_1,a_2,\ldots,a_n$. The sequential coloring algorithm with this ordering is precisely the Curtis, Powell, and Reid [1974] algorithm. Note, however, that the chromatic number of G(A) is independent of the ordering of the columns, but that the coloring produced by a sequential algorithm is dependent on the ordering of the vertices. To illustrate this remark, consider bipartite graphs: A graph G = (V,E) is bipartite if and only if it is 2-colorable. Equivalently, G = (V,E) is bipartite if and only if V is the union of two disjoint sets V_1 and V_2 such that any edge has one endpoint in V_1 and the other in V_2.

Bipartite graphs are particularly interesting because as noted in Section 3, deciding that a graph is 3-colorable is an NP-complete problem.

Consider now the bipartite graph (due to Johnson [1974]) with vertex set

$$\{u_1,u_2,\ldots,u_n\} \cup \{v_1,v_2,\ldots,v_n\} \tag{4.1}$$

and with edge set

$$\{(u_i,v_j):\ i \neq j\}\ . \tag{4.2}$$

For n = 4, this graph is shown in Figure 4.1.

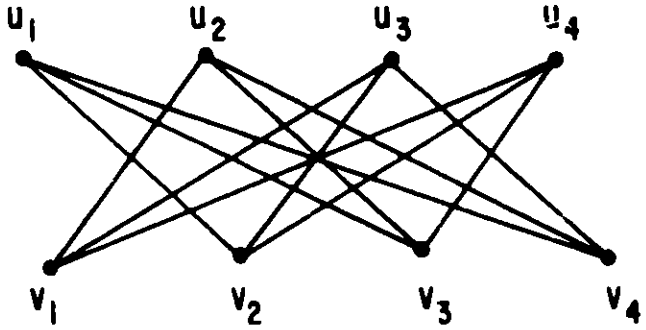

Figure 4.1

With the ordering

(4.3) $\quad u_1,v_1,u_2,v_2,\ldots,u_n,v_n$

the sequential algorithm requires n colors, but with the ordering

$$u_1,u_2,\ldots,u_n,v_1,v_2,\ldots,v_n$$

the sequential algorithm requires 2 colors.

The above example shows that the sequential algorithm with an arbitrary order may perform very poorly. On the other hand, it is not difficult to show that for any graph G there is an ordering of the vertices for which the sequential algorithm produces an optimal coloring. In fact, if ϕ is an optimal coloring of G and if the vertices are ordered so that $\{\phi(v_i)\}$ is nondecreasing, then the sequential algorithm produces an optimal coloring.

There are several techniques for ordering the vertices of G *a priori* so as to improve the performance of the sequential algorithm. The largest-first (LF) ordering of Welsh and Powell [1967] orders the vertices so that $\{d(v_i)\}$ is nonincreasing where d(v) is the degree of vertex v; that is, the number of edges with v as an endpoint. The motivation for this ordering is that the sequential algorithm produces a coloring with at most

(4.4) $\quad \max\{\min[d(v_i)+1,\ i]:\ 1 \le i \le n\}$

colors, and thus the LF ordering minimizes this bound. Note, however, that the ordering (4.3) in the bipartite graph with vertex set (4.1) and edge set (4.2) is an LF ordering and therefore the sequential algorithm with the LF ordering may still perform poorly on bipartite graphs.

The smallest-last (SL) ordering of Matula, Marble, and Isaacson [1972] is an improvement of the LF ordering. To define this ordering, assume that the vertices $v_{k+1},\ldots,v_n$ have been selected, and choose v_k so that the degree of v_k in the subgraph induced by

$$V - \{v_{k+1},\ldots,v_n\}$$

is minimal. This choice guarantees that the sequential algorithm with the SL ordering produces a coloring with at most

$$\max\{1+\delta(G_0):\ G_0 \text{ a subgraph of } G\} \tag{4.5}$$

colors where $\delta(G_0)$ is the smallest degree of G_0, and it is clear that this bound is never worse than (4.4). The sequential algorithm with the SL ordering, however, may still perform poorly on bipartite graphs. To show this consider the graph with vertex set

$$\{u_i,v_i,p_i,q_i,r_i,s_i:\ 1 \le i \le n\}\ ,$$

and whose edge set is the union of the following three sets:

$$\{(u_i,v_j):\ i \ne j,\ \ i,j = 1,2,\ldots,n\}\ ,$$
$$\{(u_i,p_j),(v_i,q_j):\ j \ge i,\ \ i,j = 1,2,\ldots,n\}\ ,$$
$$\{(r_i,p_j),\ (s_i,q_j):\ \ i,j = 1,2,\ldots,n\}\ .$$

For n = 3 this graph is shown in Figure 4.2.

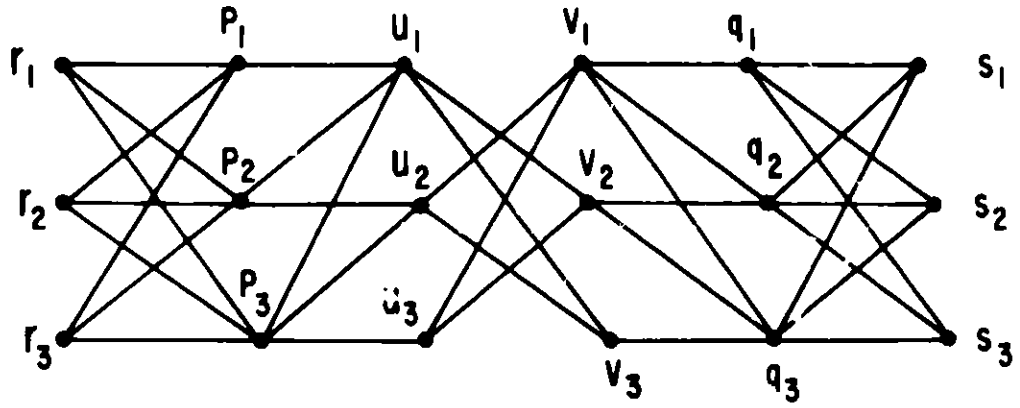

Figure 4.2

It is not difficult to show that the ordering

$$q_1,s_1,\ldots,q_n,s_n, \quad p_1,r_1,\ldots,p_n,r_n, \quad u_1,v_1,\ldots,u_n,v_n$$

is an SL ordering and that the sequential algorithm produces a coloring ϕ such that

$$\phi(p_i) = \phi(q_i) = 1 ,$$
$$\phi(r_i) = \phi(s_i) = 2 ,$$
$$\phi(u_i) = \phi(v_i) = i+1 .$$

Thus the SL ordering requires n+1 colors. On the other hand, it is clear that this graph is bipartite.

One of the interesting properties of the incidence degree (ID) ordering is that it is optimal for bipartite graphs. To define this ordering, assume that $v_1,v_2,\ldots,v_{k-1}$ have been selected and choose v_k so that the degree of v_k in the subgraph induced by

$$\{v_1,v_2,\ldots,v_k\}$$

is maximal. The incidence degree of v_k is the degree of v_k in this subgraph.

This ordering is based on the work of Brélaz [1979]. To define the algorithm proposed by Brélaz, assume that the vertices $v_1,v_2,\ldots,v_{k-1}$ have been assigned to colors, and for any uncolored vertex v, define the saturation

degree of v as the number of different colors adjacent to v. The algorithm of Brélaz now chooses v_k as a vertex with maximal saturation degree and assigns v_k to the smallest possible color. The reason for using incidence degree instead of saturation degree is that the calculation of the saturation degree for G(A) requires an excessive amount of time or space.

To prove that the ID ordering is optimal for bipartite graphs, it is convenient to introduce additional graph theory terminology. A sequence of vertices

$$v_0, v_1, \ldots, v_\ell , \qquad \ell > 0 ,$$

such that v_{i-1} is adjacent to v_i for $1 \leq i \leq \ell$ is a path of length ℓ if $v_0, v_1, \ldots, v_\ell$ are distinct, and it is a cycle of length ℓ if $v_1, v_2, \ldots, v_\ell$ are distinct and $v_0 = v_\ell$. We also need to note that if G is a bipartite graph then it has no cycle of odd length. In fact, if $v_0, v_1, \ldots, v_\ell$ is a cycle in G, and if ϕ is an optimal coloring of G, then

$$\phi(v_0) = \phi(v_2) = \phi(v_4) = \ldots$$

and since $v_0 = v_\ell$, it follows that ℓ must be even.

Theorem 4.1. The sequential algorithm with the ID ordering is optimal for bipartite graphs.

Proof: Assume that the sequential algorithm with the ID ordering has used color 3 on a bipartite graph, and let w be the first such vertex. By the definition of the algorithm, w must be adjacent to vertices w_1 and w_2, colored 1 and 2, respectively. Moreover, w_1 and w_2 must be connected by a path

$$w_1 = v_0, v_1, \ldots, v_\ell = w_2$$

of vertices colored 1 or 2. Since w_1 is colored 1, it follows that $v_0, v_2, v_4, \ldots$ are also colored 1, and since w_2 is colored 2, this implies that the length ℓ of the path must be odd. Hence,

$$w, v_0, v_1, \ldots, v_\ell, w$$

is a cycle of odd length, but this is not possible in a bipartite graph. This contradiction concludes the proof.

Although the sequential algorithm with the ID ordering is optimal for bipartite graphs, it may fail miserably on 3-colorable graphs. To illustrate this point consider the graph with vertex set

$$\{u_i, v_i, w_i: \quad i=1,2,\ldots,n\}$$

and whose vertex set consists of the union of the sets

$$\{(u_i, v_j): \quad j \leq i, \; i,j = 1,2,\ldots,n\} ,$$

$$\{(v_i, w_j): \quad i \neq j, \; i,j = 1,2,\ldots,n\} ,$$

$$\{(u_i, w_i): \quad i=1,2,\ldots,n\} .$$

For n = 4, this graph is shown in Figure 4.3.

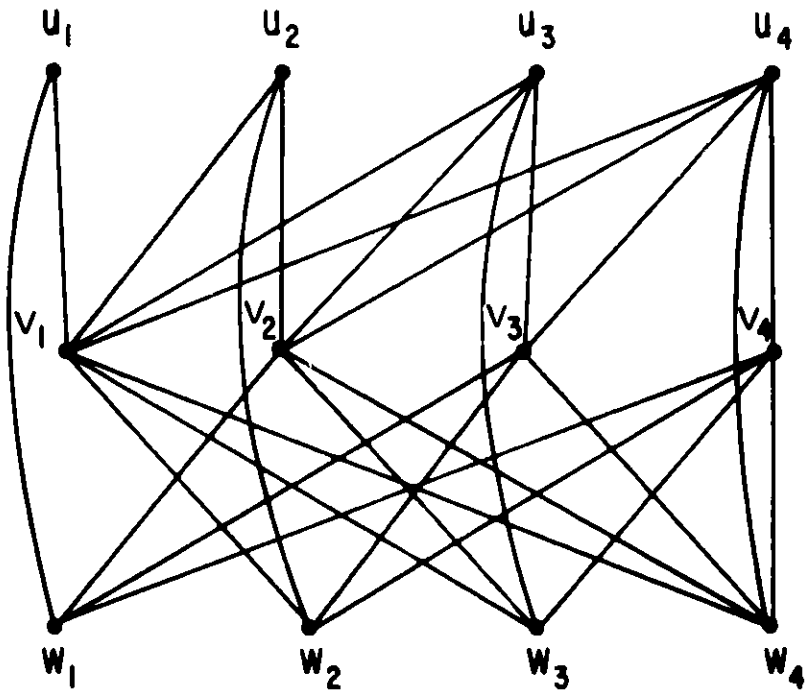

Figure 4.3

It is not difficult to show that the ordering

$$v_1,u_1,w_1,v_2,u_2,w_2,\ldots,v_n,u_n,w_n$$

is an ID ordering, and that in this case the sequential algorithm uses n colors. The chromatic number of this graph is, however, 3.

In a very interesting paper, Johnson [1974] considered thirteen coloring algorithms and showed that they all could fail miserably on 3-colorable graphs. More precisely, Johnson showed that on a graph with n vertices, twelve of the coloring algorithms may require

$$\alpha_1 n\chi(G)$$

colors for some positive constant α_1, and that for the thirteenth algorithm this bound could be improved to

$$\alpha_2[\frac{n}{\log n}]\chi(G)$$

for some positive constant α_2. It turns out that for graphs of the form G(A), this bound can be improved still further. In particular, we show below that on G(A) any reasonable coloring algorithm requires at most

$$\alpha_3 m^{1/2}\chi[G(A)] \tag{4.6}$$

colors for some positive constant α_3. This bound is optimal for the coloring algorithms that we have considered. In fact, since any graph G with n vertices can be covered by n^2 cliques, G is isomorphic to a G(A) where A is an n^2 by n matrix. In view of the examples presented in this section, it follows that the bound (4.6) is achieved for these examples.

To establish (4.6) we need a preliminary result.

Lemma 4.2. Consider any coloring algorithm for a graph G = (V,E) which produces a coloring ϕ such that every vertex v is adjacent to vertices with colors $1,2,\ldots,\phi(v)-1$. Then

$$\max\{\phi(v): \ v \in V\} \leq 1 + [2|E|]^{1/2}.$$

Proof: If $\phi(v) = \ell$ then for any colors i and j such that

$$1 \leq i < j \leq \ell$$

there is an edge (v_i, v_j) such that $\phi(v_i) = i$ and $\phi(v_j) = j$. Hence

$$\frac{\ell(\ell-1)}{2} \leq |E| ,$$

and the result is a direct consequence of this inequality.

Any reasonable coloring satisfies the assumptions of Lemma 4.2 since given any coloring ϕ_1 of a graph G, there is a coloring ϕ_2 of G which satisfies the assumptions of Lemma 4.2 and such that

$$\phi_2(v) \leq \phi_1(v) , \qquad v \in V .$$

In fact, ϕ_2 can be obtained by applying the sequential coloring algorithm to G with any ordering $v_1, v_2, \ldots, v_n$ such that $\{\phi_1(v_i)\}$ is nondecreasing.

It is now fairly easy to prove that (4.6) holds. If ρ_i is the number of nonzeroes in row i of the m by n matrix A, then the number of nonzeroes in A^TA is at most

$$\sum_{i=1}^{m} \rho_i^2 ,$$

and thus Theorem 2.2 implies that

$$2|E| \leq \sum_{i=1}^{m} \rho_i^2 \leq m \rho_{max}^2 ,$$

where

$$\rho_{max} = \max\{\rho_i : 1 \leq i \leq m\} .$$

Lemma 3.1 now shows that any coloring algorithm which satisfies the assumptions of Lemma 4.2 requires at most

$$1 + m^{1/2} \rho_{max} \leq 1 + m^{1/2} \chi[G(A)]$$

colors, and this establishes (4.6).

5. Band Problems

We have examined the behavior of several coloring algorithms and have noted that they all may perform poorly on 3-colorable graphs. In this section we examine the behavior of these coloring algorithms on graphs of the form G(A) where A is a band matrix.

The m by n matrix A has lower bandwidth

$$\beta_\ell(A) = \max\{|i-j|: \ i \geq j, \ a_{ij} \neq 0\} ,$$

and upper bandwidth

$$\beta_u(A) = \max\{|i-j|: \ i \leq j, \ a_{ij} \neq 0\} .$$

The bandwidth of A is

$$\beta(A) = 1 + \beta_\ell(A) + \beta_u(A) .$$

The bandwidth of a matrix is dependent on the ordering of the rows and columns of A. On the other hand, the corresponding notion for graphs is independent of the ordering of the vertices.

Let G = (V,E) be a graph with n vertices. The bandwidth of an ordering $v_1, v_2, \ldots, v_n$ of V is

$$\beta(\{v_k\}) = \max\{|i-j|: \ (v_i,v_j) \in E\} ,$$

and the bandwidth of G is

$$\beta(G) = \min\{\beta(\{v_k\}): \ \{v_k\} \text{ an ordering of } V\} .$$

The definition of G(A) implies that a_i and a_j are not adjacent in G(A) whenever $|i-j| \geq \beta(A)$. Thus

$$\beta[G(A)] \leq \beta[A]-1 ,$$

and equality holds if, for example, A is dense within the band. Note that it may be possible to choose permutation matrices P and Q such that

$$\beta[PAQ] < \beta(A) .$$

However, since G(PAQ) is isomorphic to G(A),

$$\beta[G(A)] \leq \beta[PAQ]-1$$

for all permutation matrices P and Q. Strict inequality may hold for all choices of P and Q. For example, if A is of the form

$$A = \begin{pmatrix} D & z \\ 0 & \alpha \end{pmatrix}$$

where z is an n-vector with nonzero components, D is a nonsingular diagonal matrix of order n, and α is a nonzero scalar, then

$$\beta[PAQ] = n+1$$

for all permutation matrices P and Q, but

$$\beta[G(A)] = \left\lfloor \frac{n+1}{2} \right\rfloor .$$

Also note that this example shows that the chromatic number of G(A) may be much smaller than the bandwidth of G(A). On the other hand, we show below that this is not the case if G(A) is a band graph.

Definition. Let G = (V,E) be a graph with n vertices. G is a band graph if there is an ordering $v_1,v_2,\ldots,v_n$ of the vertices of G such that

$$|i-j| \leq \beta(G) \iff (v_i,v_j) \in E .$$

The order $v_1,v_2,\ldots,v_n$ is a natural ordering of the band graph.

One of the main reasons for introducing band graphs is that if A is dense within the band then G(A) is a band graph. Also note that given any graph G = (V,E), we can add edges and obtain a band graph with the bandwidth of G. The

next result shows, in particular, that the chromatic number of a band graph is determined by the bandwidth of the graph.

Theorem 5.1. For any graph G,

$$\chi(G) \leq \beta(G)+1 \ .$$

Equality holds if G is a band graph.

Proof: Let $v_1,v_2,\ldots,v_n$ be an ordering of V with bandwidth $\beta \equiv \beta(G)$ and let $\ell = \beta+1$. If the sequential coloring algorithm uses the ordering

$$v_i,v_{i+\ell},v_{i+2\ell},\cdots \ , \qquad i=1,2,\ldots,\ell \ ,$$

then it requires at most ℓ colors and hence $\chi(G)$ is at most $\beta+1$. To show that if G is a band graph then $\chi(G)$ is $\beta+1$, just note that if $v_1,v_2,\ldots,v_n$ is a natural ordering of the vertices of G then

$$v_1,v_2,\ldots,v_{\beta+1}$$

are pairwise adjacent and hence $\chi(G)$ is at least $\beta+1$.

The inequality in Theorem 5.1 is due to Papadimitriou [1976]. Strict inequality may hold if, for example, G is a star graph; that is, a graph with vertex set $\{v_1,v_2,\ldots,v_n\}$ and edge set

$$E = \{(v_1,v_2),(v_1,v_3),\ldots,(v_1,v_n)\} \ .$$

In general, it is easy to show that if Δ is the maximum degree of a graph G then

$$\left\lfloor \frac{\Delta+1}{2} \right\rfloor \leq \beta(G) \ .$$

The next result shows that the sequential algorithm may perform poorly on band graphs.

Theorem 5.2. If G is a band graph then there is an ordering of the vertices of G for which the sequential algorithm requires $2\beta(G)+1$ colors.

Proof: Let $v_1,v_2,\ldots,v_n$ be a natural ordering of a band graph and set $\ell = 2\beta(G)+1$. If the sequential algorithm uses the ordering

$$v_i,v_{i+\ell},v_{i+2\ell},\ldots, \qquad i=1,2,\ldots,\ell ,$$

then it is straightforward to verify that the vertices $v_i,v_{i+\ell},v_{i+2\ell},\ldots$ are assigned to color i. Hence, the sequential algorithm requires $2\beta(G)+1$ colors.

A slight modification of the proof of Theorem 5.2 shows that the sequential algorithm with the LF ordering may also require $2\beta(G)+1$ colors. On the other hand, we now show that the sequential algorithm with the SL and ID orderings are optimal for band graphs.

Theorem 5.3. The sequential algorithm with the SL ordering requires at most $\beta(G)+1$ colors.

Proof: In Section 4 we noted that the sequential coloring algorithm with the SL ordering requires at most

$$\max\{1+\delta(G_0): \ G_0 \text{ a subgraph of } G\}$$

colors, so it is sufficient to show that $\delta(G_0) \leq \beta(G)$ for any subgraph G_0 of G. To show this, let $v_1,v_2,\ldots,v_n$ be an ordering of the vertices of G with bandwidth $\beta(G)$, and if G_0 has vertex set V_0, let

$$\ell = \min\{j: \ v_j \in V_0\} .$$

It follows that v_ℓ is adjacent to at most $\beta(G)$ vertices in G_0 and hence, $\delta(G_0) \leq \beta(G)$ as required.

Since the chromatic number of a band graph is $\beta(G)+1$, Theorem 5.3 shows that the sequential algorithm with the SL ordering is optimal for band graphs.

Theorem 5.4. The sequential algorithm with the ID ordering is optimal for band graphs.

Proof: Let G be a band graph with bandwidth $\beta = \beta(G)$ and let $v_1, v_2, \ldots, v_n$ be a natural ordering of the vertices of G. The sequential algorithm with the ID ordering first assigns colors to the vertices of a maximal clique G_0 of G, and thus the vertex set V_0 of G_0 is of the form

$$\{v_i: \ \ell \leq i \leq \ell+\beta\}$$

for some integer ℓ. It now follows that at the k-th stage of the algorithm there is an integer i_k such that the colored vertices are of the form

$$\{v_i: \ i_k \leq i \leq i_k+k\}$$

provided $k > \beta$, and that if vertices v_j and $v_{j+\beta+1}$ have been colored then v_j and $v_{j+\beta+1}$ have the same color. Hence, the incidence ordering only uses $\beta+1$ colors and is thus optimal.

Theorem 5.4 is weaker than Theorem 5.3 and this gives the SL ordering a theoretical advantage over the ID ordering. We conjecture, however, that Theorem 5.3 is also valid for the ID ordering.

6. Data Structures

The purpose of this section is to present a data structure which allows implementation of the sequential algorithms for G(A) in time proportional to

$$\sigma = \sum_{i=1}^{m} \rho_i^2$$

where ρ_i is the number of nonzeroes in row i of the m by n matrix A. This bound is satisfactory because many computations with sparse matrices require σ operations. For example, the computation of A^TA can be done by noting that

$$A^TA = \sum_{i=1}^{m} r_i r_i^T ,$$

where r_i is the i^{th} row of A, and hence requires

$$\frac{1}{2} \sum_{i=1}^{m} \rho_i(\rho_i+1)$$

operations.

We assume that the ordered pairs (i,j) for which $a_{ij} \neq 0$ are provided by two arrays indrow and indcol. Thus, if τ is the number of nonzero elements in A then

$$(\text{indrow}(k),\text{indcol}(k)) , \qquad k=1,2,\ldots,\tau ,$$

are the required pairs.

Given the arrays indrow and indcol, it is possible to sort these arrays (in time proportional to τ) and define two arrays, ipntr and jpntr, so that the row indices for column j are

$$\{\text{indrow}(k): \ \text{jpntr}(j) \leq k < \text{jpntr}(j+1)\} ,$$

and the column indices for row i are

$$\{\text{indcol}(k): \ \text{ipntr}(i) \leq k < \text{ipntr}(i+1)\} .$$

These four arrays define the data structure. Note that the sorting of indrow and indcol necessary to define ipntr and jpntr destroys the relative ordering of indrow and indcol, and now i = indrow(k), j = indcol(k) may not correspond to a nonzero entry in A.

The arrays indrow and jpntr (indcol and ipntr) provide a column-oriented (row-oriented) definition of the sparsity pattern of A. It is evident that given either definition it is possible to obtain the other definition of the sparsity pattern in time proportional to τ. Thus, we can automatically generate the data structure given any of the following three pairs of arrays: indrow and indcol, indrow and jpntr, indcol and ipntr.

The most time-consuming operation in the coloring algorithms for G(A) is to determine the vertices adjacent to a given vertex. If a_k is the given vertex, note that a_j is adjacent to a_k in G(A) if and only if there is an

index i such that $(i,j) \in S_k$ where

$$S_k = \{(i,j): \ a_{ik} \neq 0, \ a_{ij} \neq 0\} .$$

The program segment below generates S_k for k=1,2,...,n.

```
      DO 50 K = 1, N
         JPL = JPNTR(K)
         JPU = JPNTR(K+1) - 1
         IF (JPU .LT. JPL) GO TO 40
         DO 30 JP = JPL, JPU
            IR = INDROW(JP)
            IPL = IPNTR(IR)
            IPU = IPNTR(IR+1) - 1
            IF (IPU .LT. IPL) GO TO 20
            DO 10 IP = IPL, IPU
               IC = INDCOL(IP)
10          CONTINUE
20       CONTINUE
30       CONTINUE
40    CONTINUE
50    CONTINUE
```

Since each (i,j) such that $a_{ij} \neq 0$ belongs to ρ_i sets S_k, this program segment executes in time proportional to σ.

The above discussion indicates that it is possible to implement the coloring algorithms for G(A) so that they execute in time proportional to σ. The details of these implementations, however, can be quite subtle and will be described elsewhere.

7. Numerical Results

In this section we present some of the results obtained with the coloring algorithms of Section 4 on a variety of problems of the form G(A). We consider four algorithms:

CPR. A sequential algorithm with the ordering $a_1, a_2, \ldots, a_n$.

LFO. A sequential algorithm with the LF ordering.

SLO. A sequential algorithm with the SL ordering.

IDO. A sequential algorithm with the ID ordering.

Given the matrix A, the coloring produced by CPR is well-defined, but with the other algorithms, ties may affect the coloring produced. Ties were broken arbitrarily in LFO and SLO, but in IDO it was helpful to break ties by scanning the list of vertices with maximal incidence degree and choosing a vertex (column) with the least number of nonzero elements.

For each problem we cite at least three statistics. These are the matrix density (MATD) of A, the graph density (GPHD) of G(A), and the maximum number of nonzeroes in any row (MAXR) of A. If τ is the number of nonzeroes in A, then the matrix density is

$$\frac{\tau}{m \cdot n} ,$$

and if E is the set of edges of G(A) then the graph density of G(A) is

$$\frac{|E|}{\left[\frac{n(n-1)}{2}\right]} .$$

Also recall that at the beginning of Section 2 we proved that MAXR is a lower bound on the number of evaluations of Ad needed to determine A by any method.

In some of the problems we generated sequences of uniformly distributed random numbers with the RAND function of L. Schrage [1979]. Given an integer seed, RAND generates a random number in (0,1) and changes the seed. Thus a sequence of random numbers can be generated by repeated calls to RAND.

We now consider the methods used to generate the sparsity patterns of the matrices. Note that it is only necessary to specify the pattern

$$S = \{(i,j): \ a_{ij} \neq 0\}$$

and that the numerical values of a_{ij} are unimportant.

Random Matrices

We first tested the algorithms on matrices with a random sparsity pattern. The performance of the algorithms on these problems may not be

indicative of their performance in practical problems, but these problems may uncover undesirable behavior in the algorithms.

Given a density μ in [0,1], we define a random sparsity pattern S as follows:

```
Fo. j=1,2,...,n
   For i=1,2,...,m
      Generate a random number r in (0,1)
      If r ≤ μ then (i,j) ∈ S.
```

For each triple (m,n,μ) we generated 5 random patterns and averaged the results. Table 1 summarizes the results for densities 0.005, 0.01, 0.02, 0.03, and 0.04. These densities are representative of those found in applications.

The results in Table 1 indicate that with the exception of CPR, all algorithms perform similarly. CPR colors the graph in a random order, and Table 1 shows that strategic orderings are worthwhile.

Also note that on these problems the density of the graph is much larger than the density of the matrix. To explain this note that

$$\text{Prob}\{(a_i,a_j) \in E\} = 1 - (1-\mu^2)^m$$

and that for $m\mu^2 \leq 1$,

$$1 - (1-\mu^2)^m \geq 0.5m\mu^2 .$$

Thus the ratio of the density of the graph to the density of the matrix is at least $0.5m\mu$. Finally, note that although MAXR is a lower bound on the chromatic number of G(A), this bound does not seem to be very sharp for these problems.

TABLE 1

Random Matrices

M	N	MATD(%)	GPHD(%)	MAXR	CPR	LFO	SLO	IDO
500	500	0.50	1.27	8.0	9.0	8.0	8.0	8.0
500	500	1.00	4.78	12.2	15.2	13.2	12.6	13.2
500	500	2.00	18.11	22.0	34.8	29.2	30.0	30.2
500	500	2.99	36.04	27.8	59.4	51.6	52.0	52.2
500	500	3.98	54.87	34.6	88.6	78.2	79.2	79.2
1000	250	0.52	2.71	6.2	7.4	6.4	6.4	6.4
1000	250	1.00	9.64	8.8	14.4	11.8	11.8	12.2
1000	250	2.00	32.99	13.2	32.0	27.4	28.2	28.2
1000	250	3.01	59.84	17.8	54.4	48.8	49.0	50.4
1000	250	4.01	80.03	21.4	81.4	74.4	75.4	77.2
2000	125	0.50	4.97	4.6	6.2	5.2	5.0	5.0
2000	125	1.00	18.18	6.8	13.0	11.0	10.8	11.6
2000	125	2.01	55.23	9.4	28.6	26.0	26.4	26.4
2000	125	2.99	83.07	11.4	48.6	44.8	45.2	46.6
2000	125	3.99	95.73	13.8	69.8	65.6	66.6	68.8

Band Matrices

One of the reasons why the results for general random matrices may not be indicative of their behavior in practice is that the density of the graph is large relative to the density of the matrix. It is possible to generate graphs G(A) with a smaller relative density by considering band matrices.

The idea is to generate a permutation of a band matrix with bandwidth $2\beta+1$. This can be done by first obtaining a random permutation $\pi(1),\pi(2),\ldots,\pi(n)$ of the integers $1,2,\ldots,n$, and then an appropriate sparsity pattern S can be generated as follows:

```
For j=1,2,...,n
   For i = max(1,π(j)-β),...,min(n,π(j)+β)
      Generate a random number r in (0,1)
      If r ≤ μ then (i,j) ∈ S.
```

In this construction μ is the density within the band. For each triple (n,β,μ) we generated 5 patterns and averaged the results. Table 2 summarizes

the results for n = 500, μ = 1.0,0.5,0.25, and β = 4,8,16,32. These settings lead to reasonable matrix densities.

The results in Table 2 show that on these problems CPR continues to perform poorly. LFO performs better than CPR but SLO and IDO perform better still. Also note that if the density of the graph is not too much larger than the density of the matrix, then the lower bound provided by MAXR is fairly sharp.

TABLE 2

Band Matrices (n=500)

β	MATD(%)	GPHD(%)	MAXR	CPR	LFO	SLO	IDO
4	1.79	3.18	9.0	13.6	13.8	9.0	9.0
4	0.90	2.12	8.4	10.8	9.8	8.8	8.8
4	0.45	0.80	6.6	7.0	6.8	6.8	6.8
8	3.37	6.30	17.0	25.6	25.0	17.0	17.0
8	1.70	5.16	14.4	20.2	19.4	16.6	16.4
8	0.83	2.49	10.4	12.0	10.8	10.4	10.6
16	6.49	12.40	33.0	47.4	46.8	33.0	33.0
16	3.27	11.28	25.6	41.4	39.2	32.6	32.6
16	1.62	7.40	16.4	24.2	21.6	19.6	20.2
32	12.58	23.98	65.0	86.8	87.2	65.0	65.0
32	6.28	22.94	44.6	84.0	81.4	64.0	64.2
32	3.14	18.65	26.6	55.2	52.0	44.0	44.2

Naval Problems

Our third test consists of the matrices described by Everstine [1979] of the David W. Taylor Naval Ship Research and Development Center. These problems were obtained from users representing various Navy, Army, Air Force and NASA laboratories. Table 3 summarizes the results obtained by the algorithms; to compare the overall performance of the algorithms on this problem set, we have included below the total number of colors required by the algorithms.

MAXR	CPR	LFO	SLO	IDO
408	473	461	433	435

The obvious conclusion is that SLO and IDO perform best overall. It is particularly interesting that the difference between the lower bound MAXR and the results for SLO and IDO is on the average less than 1 color. Since MAXR is a lower bound on the number of evaluations of Ad needed to determine A by any method, this shows that these algorithms are nearly optimal.

TABLE 3

Naval Problems

NP	N	MATD(%)	GPHD(%)	MAXR	CPR	LFO	SLO	IDO
1	59	7.67	14.96	6	8	7	6	6
2	66	7.35	11.89	6	6	6	6	6
3	72	4.28	6.65	5	5	5	5	5
4	87	7.15	19.41	13	16	13	13	13
5	162	4.50	9.96	9	11	12	10	10
6	193	9.38	30.65	30	31	31	32	32
7	198	3.55	7.80	12	12	12	12	12
8	209	3.99	11.91	17	17	17	17	17
9	221	3.34	8.21	12	14	13	12	13
10	234	1.52	3.65	10	10	10	10	10
11	245	2.43	6.81	13	16	14	13	13
12	307	2.68	6.85	9	13	14	11	12
13	310	2.55	6.18	11	12	13	11	11
14	346	2.69	10.08	19	24	22	20	21
15	361	2.27	5.66	9	12	10	11	10
16	419	2.03	5.57	13	17	17	15	15
17	492	1.30	3.17	11	13	12	11	11
18	503	2.38	7.10	25	28	28	25	25
19	512	1.34	3.38	15	18	16	16	16
20	592	1.46	3.86	15	17	17	15	15
21	607	1.39	4.08	14	18	18	17	17
22	758	1.04	2.49	11	13	15	12	12
23	869	.96	2.46	14	16	16	15	15
24	878	.97	2.45	10	11	12	11	11
25	918	.88	2.31	13	16	18	14	14
26	992	1.70	4.39	18	18	18	22	20
27	1005	.85	2.54	27	32	27	27	27
28	1007	.85	2.16	10	11	12	11	12
29	1242	.68	1.87	12	17	15	14	15
30	2680	.35	1.03	19	21	21	19	19

All of the matrices in the naval problems are symmetric but the coloring algorithms ignore this feature; algorithms for determining sparse symmetric matrices will be discussed in another paper. To test the dependence of the algorithms on the ordering of the columns and to obtain a nonsymmetric set of test matrices, we permuted the columns of the naval matrices. The permuted naval problems were obtained by considering the columns in the same order as in the tape provided by Everstine -- it turns out that this leads to non-symmetric matrices. Table 4 summarizes the results obtained by the algorithms.

The most striking feature of the results in Table 4 is that CPR is so dependent on the ordering of the columns while the other three algorithms are relatively insensitive. This can be seen at a glance by comparing the total number of colors required by the algorithms in each case. The totals for Table 4 are

MAXR	CPR	LFO	SLO	IDO
408	488	462	434	434

and these results show that on the permuted naval problems SLO and IDO again perform best overall.

TABLE 4
Permuted Naval Problems

NP	N	MATD(%)	GPHD(%)	MAXR	CPR	LFO	SLO	IDO
1	59	7.67	14.96	6	8	7	6	6
2	66	7.35	11.89	6	6	6	6	6
3	72	4.28	6.65	5	6	5	5	5
4	87	7.15	19.41	13	13	13	13	13
5	162	4.50	9.96	9	11	12	9	9
6	193	9.38	30.65	30	33	30	31	32
7	198	3.55	7.80	12	15	12	12	12
8	209	3.99	11.91	17	17	17	17	17
9	221	3.34	8.21	12	13	13	12	13
10	234	1.52	3.65	10	10	10	10	10
11	245	2.43	6.81	13	14	14	13	13
12	307	2.68	6.85	9	15	14	12	13
13	310	2.55	6.18	11	13	14	11	11
14	346	2.69	10.08	19	24	22	21	21
15	361	2.27	5.66	9	12	10	10	10
16	419	2.03	5.57	13	18	17	15	16
17	492	1.30	3.17	11	12	12	11	11
18	503	2.38	7.10	25	32	29	25	25
19	512	1.34	3.38	15	20	17	18	16
20	592	1.46	3.86	15	17	16	15	16
21	607	1.39	4.08	14	20	18	17	17
22	758	1.04	2.49	11	12	15	11	11
23	869	.96	2.46	14	16	16	15	15
24	878	.97	2.45	10	12	12	11	11
25	918	.88	2.31	13	16	18	14	14
26	992	1.70	4.39	18	18	18	22	20
27	1005	.85	2.54	27	33	28	27	27
28	1007	.85	2.16	10	11	12	11	11
29	1242	.68	1.87	12	17	15	15	14
30	2680	.35	1.03	19	24	20	19	19

Harwell Problems

The fifth test consists of the matrices described by Duff and Reid [1979] of the Harwell Atomic Energy Research Establishment. There are 36 matrices in this collection, but matrices 33 through 36 have the same pattern, so only 33 matrices are used in our results. With the exception of matrices 1,2,3,9,10, all matrices in this collection are unsymmetric. Moreover, matrices 28,29,30, 31,32 are not square; their dimensions (m,n) are (219,85), (958,292), (331,104), (608,188), and (313,176), respectively. Table 5 summarizes the results obtained by the algorithms, and we have included below the total

number of colors required by the algorithms.

MAXR	CPR	LFO	SLO	IDO
4473	4530	4512	4500	4495

It is clear that on the Harwell problems algorithms SLO and IDO perform better than CPR and LFO, and that SLO and IDO are nearly optimal.

TABLE 5

Harwell Problems

NP	N	MATD(%)	GPHD(%)	MAXR	CPR	LFO	SLO	IDO
1	147	11.33	26.44	21	28	27	24	22
2	147	11.30	26.44	21	28	27	24	22
3	1176	1.34	3.39	99	107	99	99	99
4	113	5.13	17.68	20	21	20	20	20
5	32	12.30	36.09	8	9	8	8	8
6	54	9.98	23.55	12	12	12	12	12
7	57	8.65	19.05	11	11	11	11	11
8	199	1.77	4.13	6	9	8	7	7
9	292	2.59	6.77	14	16	16	14	14
10	85	7.24	17.25	10	11	10	10	10
11	130	7.59	92.58	124	124	124	124	124
12	663	.38	40.67	422	422	422	422	422
13	663	.39	44.21	440	440	440	440	440
14	663	.39	41.45	426	426	426	426	426
15	363	1.86	5.14	34	35	34	34	34
16	363	2.33	6.06	30	31	30	30	30
17	363	2.40	6.09	33	34	33	33	33
18	363	2.49	6.47	33	33	33	33	33
19	822	.48	11.34	266	266	266	266	266
20	822	.56	12.95	283	283	283	283	283
21	822	.60	14.09	295	295	295	295	295
22	822	.62	14.77	302	302	302	302	302
23	822	.67	15.28	304	304	304	304	304
24	822	.69	15.79	308	308	308	308	308
25	822	.70	16.06	311	311	311	311	311
26	822	.71	16.15	311	311	311	311	311
27	822	.72	15.64	304	304	304	304	304
28	85	2.35	6.13	2	5	5	4	4
29	292	.68	2.25	2	7	6	6	6
30	104	1.92	6.18	2	6	6	6	6
31	188	1.06	3.46	2	6	6	6	6
32	176	2.83	8.50	6	10	11	10	10
33	541	1.46	4.61	11	15	14	13	12

8. Concluding Remarks

The numerical results of Section 7 show that the problem of evaluating a sparse Jacobian matrix can be attacked quite successfully as a graph coloring problem, and moreover, that the SL and ID orderings are nearly optimal for practical problems. In spite of their excellent performance, in some cases it may be possible to improve the performance of these algorithms. Consider, for example, the density pattern of a block tridiagonal matrix whose diagonal blocks are also tridiagonal and whose off-diagonal blocks are diagonal. Melgaard and Sincovec [1981] have shown that matrices with this pattern can be estimated (optimally) with 5 evaluations of Ad. It turns out, however, that our algorithms are not optimal for this pattern, although for the SL ordering it can be shown that in the worst possible case it requires 7 evaluations of Ad.

It may also be possible to improve on the graph coloring algorithms by using a hybrid approach. To be more specific, consider problem 11 of the Harwell collection -- the laser matrix. This matrix has one dense row with 124 nonzero elements, but the remaining rows have at most 58 nonzero elements. Our algorithms require 124 evaluations of Ad to determine the laser matrix A and this is optimal. However, if we could estimate the dense row separately, then the remaining problem only requires 58 evaluations of Ad. The success of this approach depends on the cost of estimating the dense row. If (as assumed in the introduction) it is more efficient to evaluate $F(x)$ than to evaluate the components $f_1(x),f_2(x),\ldots,f_m(x)$ of $F(x)$ separately, then this approach may not help, but if the component of F corresponding to the dense row is relatively easy to compute then a hybrid approach would be very helpful.

Acknowledgments

This work benefitted from interactions with many people. Julian Araoz was the first to note the connection between estimation of sparse Jacobian matrices and the graph G(A); since then many people have told us that they were aware of this connection but were unable (for various reasons) to pursue it further. Ed Dean spent a summer implementing and testing variations of the coloring algorithms. Gordon Everstine and Iain Duff kindly supplied us with tapes of their sparse matrix collections. Ken Hillstrom provided valuable assistance and, in particular, plots of the sparse matrices. Finally, Judy Beumer typed this manuscript with incredible speed and accuracy.

References

Bondy, J. A., and Murty, U. S. R. [1976]. Graph Theory with Applications, North Holland.

Brélaz, D. [1979]. New methods to color the vertices of a graph, Comm. ACM 22, 251-256.

Curtis, A. R., Powell, M. J. D., and Reid, J. K. [1974]. On the estimation of sparse Jacobian matrices, J. Inst. Math. Appl. 13, 117-119.

Duff, I. S. and Reid, J. K. [1979]. Performance evaluation of codes for sparse matrix problems, in Performance Evaluation of Numerical Software, L. D. Fosdick, ed., 121-135, North Holland.

Eisenstat, S. [1980]. Personal communication.

Everstine, G. C. [1979]. A comparison of three resequencing algorithms for the reduction of matrix profile and wave front, Internat. J. Numer. Methods Engrg. 14, 837-853.

Garey, M. R. and Johnson, D. S. [1979]. Computers and Intractability, W. H. Freeman.

Harary, F. [1969]. Graph Theory, Addison-Wesley.

Johnson, D. S. [1974]. Worst case behavior of graph coloring algorithms, in Proceedings 5th Southeastern Conference on Combinatorics, Graph Theory, and Computing, 513-527, Utilitas Mathematica Publishing.

Matula, D. W., Marble, G., and Isaacson, J. D. [1972]. Graph coloring algorithms, in Graph Theory and Computing, R. C. Read, ed., 104-122, Academic Press.

Melgaard, D. K. and Sincovec, R. F. [1981]. General software for two-dimensional nonlinear partial differential equations, ACM Trans. Math. Software 7, 106-125.

Moon, J. and Moser, L. [1965]. On cliques in graphs, Israel J. Math. 3, 23-28.

Papadimitriou, Ch. [1976]. The NP-completeness of the bandwidth minimization problem, Computing 16, 263-270.

Powell, M. J. D. and Toint, P. L. [1979]. On the estimation of sparse Hessian matrices, SIAM J. Numer. Anal. 16, 1060-1074.

Schrage, L. [1979]. A more portable Fortran random number generator, ACM Trans. Math. Software 5, 132-138.

Welsch, D. J. A. and Powell, M. B. [1967]. An upper bound for the chromatic number of a graph and its application to timetabling problems, Comput. J. 10, 85-86.